Selecting an Optimum Configuration of One-Way and Two-Way Routes Using Tabu Search

AUTHOR(S):

Drezner, Zvi; Salhi, Said

CITATION:

Drezner, Zvi ...[et al]. Selecting an Optimum Configuration of One-Way and Two-Way Routes Using Tabu Search. 数理解析研究所講究録 1998, 1068: 203-215

ISSUE DATE:

1998-10

URL:

http://hdl.handle.net/2433/62510

数理解析研究所講究録
1068巻 1998年 203-215

Selecting an Optimum Configuration of One-Way and Two-Way Routes Using Tabu Search

Zvi Drezner
Department of Management Science/Information Systems
California State University-Fullerton
Fullerton, CA 92834.

Said Salhi
Management Mathematics Group
School of Mathematics and Statistics
University of Birmingham
Birmingham, United Kingdom.

Abstract

The problem of designing a near optimal configuration of a system of one-way and two way route system is investigated. Each arc of the network can be designed as either a two-way arc or a one-way arc in one of the two directions. The traffic speed on a one-way arc is faster than the speed on a two-way arc by a given factor. The problem is to design a network which minimizes total travel time between all pairs of nodes by the proper selection of one-way and two-way arcs. Efficient implementations of the metaheuristic Tabu search are designed for solving this network design problem. These approaches are tested on a set of network problems with encouraging results.

1 Introduction

In recent years, network design problems have received increasing attention, see [1, 3]. Network design problems arise in several applications varying from industrial installations requiring the transport of materials to communication networks. In this paper we address the problem of designing optimum configurations for one- and two-way routes system. A network of roads is given and each road can be designed either as a two-way street or a one-way street in one direction. The objective is to optimize the configuration of one-way streets in a network for a given efficiency ratio for one-way travel. Making streets or routes one way is well known to increase the capacity of the artery, while at the same time increasing the travel distances for some flows. Many congested urban areas, for example, employ one-way streets. One-way traffic is

also useful in many industrial installations requiring the transport of materials. Communication networks also use one-way flows. The model can be applied in a wide variety of congested traffic, shop-floor flow and communication situations. Drezner and Wesolowsky [3] were the first ones to study this problem. They designed various heuristic procedures for its solution.

2 Definitions

Define:

n Number of nodes in the network

m Number of arcs in the network

S The set of all arcs. Each member in S is defined as an ordered pair (i, j) of nodes. Each arc is defined twice in both directions. The cardinality of S is $2m$.

d_{ij} defined for all $(i, j) \in S$, and is the length of the arc (i, j) when it is a two-way street. It is possible that $d_{ij} \neq d_{ji}$.

α is the factor by which distances are multiplied when an arc is turned into a one-way street ($\alpha < 1$). $\frac{1}{\alpha}$ is the factor by which the speed is increased when a two-way street is turned into a one way street.

w_{ij} is the number of vehicles traveling from origin i to destination j.

W is the set of all origin-destination pairs (i, j) for which $w_{ij} > 0$.

z_{ij} for $(i, j) \in S$ is a variable defining a particular one-way configuration for the network. $z_{ij} = 0$ if the arc from i to j is not available, and $z_{ij} = 1$ if the arc is available.

Z is the vector of variables z_{ij} of length $2m$.

$s_{ij}(Z)$ is the shortest distance from i to j for a network defined by Z.

$F(Z)$ is the objective function for a given Z.

$$F(Z) = \sum_{(i,j)\in W} w_{ij} s_{ij}(Z). \tag{1}$$

An efficient procedure for the calculation of $F(Z)$ is detailed in [3].

Various heuristic approaches of the greedy type are studied in [3]. The recommended procedure is termed there as "the modified algorithm". In this paper we refer to it as the "greedy" heuristic. The main steps of the greedy heuristic are as follows:

The Greedy Heuristic

1. The starting solution is all two-way streets.
2. Each arc i - j can assume three values: two way street, one way from i to j, and one way from j to i. The direction of each arc can be changed in two ways. There are therefore $2m$ individual changes in arc directions. The $2m$ possible changes are checked starting from a randomly selected arc. Arcs are checked consecutively in a circular way until the arc preceding the one we started with is reached.
3. Each time a better solution is found, it is accepted as the next solution and a new iteration begins.
4. If none of the $2m$ possible changes produces a better solution, the algorithm terminates.

3 Tabu Search Methods

Tabu Search methods were proposed by Glover [4]. A review of Tabu search methods can be found in [5, 7].

3.1 The Basic Tabu Search Procedure

3.1.1 Initialization:

- Generate an initial solution, Z.

- Set the best current solution $Z_{best} = Z$.
- Define a neighborhood $N(Z)$ and evaluate all the moves in the neighborhood.
- Define the Tabu list and set values for the Tabu size.
- Set the counter $iter = 0$ (current iteration).

3.1.2 The Tabu List

- The Tabu list contains a list of Tabu moves.
- The length of the Tabu list is termed "Tabu size".
- When a move is performed (for example, z_{ij} is changed from 0 to 1), $z_{ij} = 0$ is added to the Tabu list.
- A more efficient procedure is to record for each state (for example, $z_{ij} = 0$) the last iteration number for which it was changed from 0. A state is in the Tabu list if the difference between the current iteration number and the recorded value is not greater than the Tabu size.

3.1.3 Selection Strategy:

1. Evaluate $F(Z')$ for all $Z' \in N(Z)$ in a random order as the order in the greedy heuristic.
2. If $F(Z') < F(Z_{best})$ for any $Z' \in N(Z)$, terminate the iteration (i.e., do not proceed to check whether there are even better solutions in $N(Z)$), set $Z_{best} = Z^* = Z'$, and go to step 4.
3. Else, choose among the admissible solutions $Z' \in N(Z)$ the best one. Let the best one be Z^*. (An admissible solution is a solution whose move is not in the Tabu list.)
4. Set $Z = Z^*$ and $iter = iter + 1$.
5. Update the Tabu list and go to Step 1.

3.1.4 Stopping Criterion:

A suitable stopping criterion such as a limit on the number of iterations is required.

3.1.5 Diversification (Optional):

Apply some forms of diversification on well defined solutions. A diversification scheme that was used in this paper is outlined below in section 4.5.

These Tabu search steps seem to be straight forward to implement. However, the success of the method is dependent on having a good insight of the problem. There are a few questions which usually help in devising a successful implementation. The success of Tabu search methods depends on the choice of the Tabu size, the definition of the neighborhood, how diversification schemes are developed and employed, the way previous solutions are identified, the efficiency of the computer program, and above all a good understanding of the problem. For further details on these issues, see [5, 7].

4 Various Parameters for the Tabu Search

4.1 The Neighborhood

Since each arc i - j can assume three values: two way street, one way from i to j, and one way from j to i, there are 3^m possible feasible solutions to the problem. The neighborhood to be searched consists of $2m$ possible changes in the road structure. These consist of changing a certain arc from the direction it is in the present iteration to one of the other two possible directions.

4.2 Starting Solutions and Stopping Criterion

Two possible schemes are proposed for generating the set of starting solutions, and terminating the Tabu search. In each, the greedy heuristic is applied K times producing K terminal solutions.

To have control over the running time of the Tabu search, we adopted the following stopping rules. Note that other measures which are geared toward solution quality instead of computing time also exist. For instance, the use of non-improvement over the best solution after a permitted number of iterations, or no improvement in a certain number of successive iterations, etc.. These stopping rules, though producing good results, can be too time consuming. The following two rules require the same total number of Tabu iterations per problem and thus a similar computer time.

- Each terminal greedy solution defines a starting solution for the Tabu search. The number of Tabu iterations is set to the *same* number that were required for the greedy heuristic. The number of iterations required for the greedy heuristic is an indicator for the complexity of the problem.
- The *best* greedy solution is used as the starting solution in our Tabu search methods. The number of iterations in the Tabu search is K times the number of the iterations required for obtaining the best greedy solution.

4.3 The Tabu List

Two different schemes are proposed for handling the Tabu list:

- Whenever a new best solution Z_{best} (including the final greedy solution) is found, the Tabu list is emptied. We do not forbid previous Tabu search moves leading to the best solution. We treat Z_{best} as if it is a starting solution for the Tabu search.
- The Tabu list was maintained throughout the search including the iterations performed by the greedy algorithm preceding the start of the Tabu search.

4.4 The Tabu Size

The common way of defining the Tabu size is to set it to a fixed value a priori [6] or to choose the Tabu size from a defined range [8]. In this study we define the Tabu size in a dynamic manner

by letting the Tabu size value change at each iteration. Two ways to determine the Tabu size at each iteration are proposed:

4.4.1 Alternating Tabu Size Value

The Tabu size TS is alternated between two values TS_{min} and TS_{max} regardless of the change in the value of the objective function. It is similar, in principle, to the random selection of the Tabu size at every iteration. This is also similar in concept to the systematic dynamic Tabu tenure where the values alternatively increased then decreased within a range [5]. In our experiments the alternating approach seems to work well. In our view, this may provide researchers with another facet of Tabu search.

4.4.2 Varying the Tabu Size

- Two values constraining the Tabu size are set. These are TS_{min} and TS_{max}.
- The Tabu size, TS, in any given iteration is always bounded by $TS_{min} \leq TS \leq TS_{max}$.
- Whenever a new best solution is found, or when a diversification is performed, TS is set to TS_{max}.
- Each time the value of the objective function increases, TS is decreased by one as long as it does not go below TS_{min}.
- Each time the value of the objective function decreases, TS is increased by one as long as it does not exceed TS_{max}.

4.5 A Diversification Scheme

This module is introduced to guide the search in exploring other regions which might be difficult to visit otherwise. This is useful when the search appears not to discover any promising solution. It may not be easy to detect the right timing for diversification or to define the appropriate

way of diversifying the search. One common approach is to start from a random solution or a solution which has much dissimilarity as possible with other solutions. In this study the following diversification procedure is suggested:

- The Tabu size is varied as in section 4.4.2.
- A diversification is performed when $TS = TS_{min}$ is reached. Since TS is increased by a down move in the objective function and decreased by an up move, then, in most cases, a diversification is performed when there are $TS_{max} - TS_{min}$ more up moves than down moves since the last diversification (or a new best solution found). An exception to this rule occurs when a down move in the value of the objective function occurs when $TS = TS_{max}$. This is very rare and may result in a diversification when the difference between up and down moves is less than $TS_{max} - TS_{min}$.
- The diversification diverts the search back to Z_{best} but forcing the search to choose a different path from there. This different path is achieved by defining a forbidden list which contains the first two moves previously performed from Z_{best}. This forbidden list is augmented with new forbidden moves whenever we diversify back to the same Z_{best}.
- The first two exchanges which are chosen after either a diversification or when the last new best solution was found are put in a new list which we refer to as the forbidden list. Note that this forbidden list is different from the Tabu list. This list is emptied when a new best solution is found, initially contains the first two moves after the best new solution is found, and is increased by two moves at every diversification.
- The solution is returned to Z_{best} while stopping the search from following the same path which was previously selected when Z_{best} was initially found. This means that the moves in the forbidden list are not allowed to be used. A similar approach was successfully applied by [9] when solving the project scheduling problem with limited resources.
- After each diversification, the Tabu list is emptied except for the forbidden list.

5 The Test Problems and Computational Results

We used two scenarios to evaluate the performance of the proposed Tabu search heuristics: the medium and small test problems given by [3].

5.1 Investigational Experimentation

We tried many possible strategies in the implementation of the Tabu search. In all the investigative experimentations we solved the five test problems given in [3]. All these problems have the same network with $n = 40$ nodes and $m = 99$ arcs. The five problems differ in the value of α. The neighborhood for these problems consists of 198 possible exchanges. Evaluating the value of the objective function for each member in the neighborhood took less than 0.03 seconds on a Pentium 166MHz computer. Therefore, each iteration required less than 6 seconds of computer time. We performed the greedy algorithm $K = 8$ times. Therefore, if the best greedy solution is used for the Tabu search, the number of iterations is $K = 8$ times the number of the iterations required by the greedy heuristic. We checked all the cross possibilities of the options for starting solutions, handling the Tabu list, and determining the Tabu size, as described above.

We concentrated on four main approaches:

Fixed Using a fixed length Tabu list (TS), i.e., $TS_{min} = TS_{max}$.

Alt Alternating between TS_{min} and TS_{max} in consecutive iterations regardless of the values of the objective function in these iterations.

Div The Tabu size was varied as described in section 4.4.2 and a diversification performed as given in section 4.5.

NoDiv Varying the Tabu size as described in section 4.4.2 but without using diversification.

We concluded from these tests that:

Table 1: Results of 100 Runs of the Alternating Method

α	Best Known	Greedy						Tabu Search					
		Objective		Iterations		Time (Min.) per run		Objective		Total Iterations		Time (Min.) per run	
		Min	Avg.	Min	Avg.	Min	Avg.	Min	Avg.	Min	Avg.	Min	Avg.
0.5	109080	108560	110153	119	138	0.53	0.94	108264	109277	238	276	11.36	13.53
0.6	129122	128979	130459	105	126	0.55	0.94	128292	129322	210	251	9.51	12.14
0.7	147165	147529	148802	89	107	0.54	0.90	147012	148085	178	214	8.24	10.21
0.8	160608	160591	161112	51	66	0.53	0.77	160218	160836	102	132	4.60	6.41
0.9	166379	166379	166401	12	13	0.25	0.44	166379	166379	24	27	1.12	1.53

- Starting the Tabu search from each terminal greedy solution is superior in most cases to selecting the best greedy solution and performing the Tabu search with eight times the number of iterations.

- Emptying the Tabu list whenever a new best solution is better than retaining the history of Tabu moves in the majority of cases.

- We tested various fixed values of TS and various values for TS_{min} and TS_{max}. It was found that the values of 10 (5%) and 20 (10%) of the neighborhood size are better than other values.

- The best performance was obtained by the alternating method. A close second was the diversification approach. However most of the methods described above performed quite well.

5.1.1 Obtaining the Bench Mark Solutions

Since the optimal solutions for these problems are not known, we conducted an extensive computation to obtain high quality solutions which we refer to as our bench mark.

We performed the following experiments:

The alternating method was repeated one hundred times for each α to obtain a good solution for each case. The results are summarized in Table 1. The best solution that was found by the

Table 2: Using Additional Iterations on the Best Solution of Table 1

α	Best Known	Best Greedy	Bench Mark
0.5	109080	108560	108253†
0.6	129122	128979	128144
0.7	147165	147529	146666
0.8	160608	160591	160192
0.9	166379	166379	166379

† A better solution of 108213 was obtained by starting with the $\alpha = 0.6$ solution and performing one greedy iteration.

Tabu search in these one hundred experiments, was run an additional 1000 iterations, using the alternating methods and then 1000 more iterations using the diversification method. This extensive Tabu search resulted in the best solutions listed in Table 2. These solutions are used as the bench mark for comparison between various approaches.

5.1.2 Medium Size Problems

For comparison, five approaches were repeated ten times each (rather than eight times in the preliminary experiments). We report for each problem the best solution found in these ten experiments, and the average value of the objective function. The results are summarized in Table 3. The best value in each column is marked in boldface. These results confirm the results of the preliminary experiments. The alternating method was best (or tied for best) in six out of ten measures. The diversification method was best in five, the no-diversification approach was best three times, and the fixed approach was best in four cases each.

5.1.3 Small Problems

We experimented with the five small problems tested in [3]. This small problem contains $n = 14$ nodes, and $m = 20$ arcs. The size of the neighborhood is therefore $2m = 40$. The 5%-10% rule for the Tabu size leads to Tabu size between 2 and 4. For such a small problem these Tabu

Table 3: Ten Repetitions of Various Tabu Approaches

	$\alpha = 0.5$		$\alpha = 0.6$		$\alpha = 0.7$		$\alpha = 0.8$		$\alpha = 0.9$	
Method	Min	Avg.	Min	Avg.	Min	Avg.	Min	Avg.	Min	Avg.
Bench Mark	108213		128144		146666		160192		166379	
Alt	**108564**	109082	129047	129523	**147254**	**148020**	**160218**	160821	**166379**	**166379**
Div	108615	109163	**128718**	129433	147301	148111	**160218**	**160757**	**166379**	**166379**
NoDiv	108615	109113	128853	129545	147301	148127	**160218**	160781	**166379**	**166379**
Fixed (10)	108715	**109061**	129027	129609	**147254**	148204	160228	160822	**166379**	**166379**
Fixed (20)	108594	109091	128867	**129396**	147288	148114	**160218**	160768	**166379**	**166379**

Table 4: Results for the Small Problems

	$\alpha = 0.5$		$\alpha = 0.6$		$\alpha = 0.7$		$\alpha = 0.8$		$\alpha = 0.9$	
Desc.	†	Avg.	†	Avg.	†	Avg.	†	Avg.	†	Avg.
Optimum	1385.0		1656.6		1859.2		1973.0		1998.0	
Greedy	4	1439.4	1	1708.0	6	1880.1	**50**	**1973.0**	**50**	**1998.0**
Sim. An.(1)	5	1423.4	10	1690.4	7	1871.5	35	1974.1	48	1998.2
Sim. An.(2)	12	1419.3	6	1691.4	5	1872.1	21	1977.0	39	1999.2
Alt	17	1405.5	23	1681.3	43	1864.1	**50**	**1973.0**	**50**	**1998.0**
Div	**38**	**1390.9**	**43**	**1663.2**	**50**	**1859.2**	**50**	**1973.0**	**50**	**1998.0**
NoDiv	19	1405.2	27	1676.1	41	1864.2	**50**	**1973.0**	**50**	**1998.0**
Fixed (3)	12	1422.6	30	1679.1	11	1866.7	**50**	**1973.0**	**50**	**1998.0**
Fixed (6)	28	1396.8	36	1669.0	11	1861.2	**50**	**1973.0**	**50**	**1998.0**

† Number of optimal solutions

sizes are too small. We obtained better results by using Tabu sizes of 3 and 6. The results are summarized in Table 4. Each procedure was repeated fifty times. One run took about one second of computer time. In the table we give the optimal solution and two simulated annealing experiments that are reported in [3]. We report new results for the greedy algorithm, and the five approaches tested for the medium size problem. For these particular problems, all methods obtained the optimal solution at least once. In these small problems the diversification approach outperformed the other variants and produced, on the average, solutions within 0.8% of optimality.

6 Conclusions

This study investigates the optimal design of a road system which may include one-way and two-way routes. A Tabu search technique is developed in order to find good solutions for this problem. New ways of handling the Tabu size and a new diversification strategy are proposed. These implementations are tested on two random networks. These approaches yield better results than those obtained by commonly used strategies employed in Tabu search.

The diversification approach was found to be the best one when a relatively large number of iterations are allowed in the Tabu search. When we limit the number of iterations to a smaller number, the alternating approach performs slightly better.

References

[1] Cantrella, G.E. and A. Vitetta "A Multicriteria Analysis for Network Design and Parking Location," Reprints of the TRISTAN II, Capri, June 1994, 839-852.

[2] Drezner Z. and T. Drezner "Applied Location Theory Models," in *Modern Methods for Business Research*, G.A. Marcoulides (Ed.), LEA , Mahwah, New Jersey, 1998.

[3] Drezner, Z., and G.O. Wesolowsky (1997) "Selecting an Optimum Configuration of One-Way and Two-Way Routes," *Transportation Science*, 31, 386-394.

[4] Glover, F. (1986) "Future Paths for Integer Programming and Links to Artificial Intelligence, *Computers and Operations Research*, 13, 533-549.

[5] Glover F., and M. Laguna *Tabu Search*, Kluwer Academic Publishers, Boston, 1997.

[6] Osman, I.H., and Salhi, S. "Local Search Strategies for the Vehicle Fleet Mix Problem," in *Modern Heuristic Search Techniques* by Rayward-Smith, V.J., Osman, I.H, Reeves, C.R and Smith, G.D. (Eds) (1996) 131-154, Wiley.

[7] Salhi S. "Heuristic Search Methods," in *Modern Methods for Business Research*, G.A. Marcoulides (Ed.), LEA , Mahwah, New Jersey, 1998.

[8] Skorin-Kapov, J. (1990) "Tabu Search Applied to the Quadratic Assignment Problem, *ORSA Journal on Computing*, 2, 33-45.

[9] Thomas, P., and Salhi, S. (1998) "A Tabu Search Heuristic for the Resource Constrained Project Scheduling Problem, *Journal of Heuristics*, 4, 123-139.